"Express Mail" mailing label number EL485650226US
Date of Deposit June 1, 2000

Atty Docket No. 00 P 7661 US

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

This is a U.S. Patent Application for:

TITLE: METHOD FOR ADDING EXTENSIONS TO THE GRAMMAR FOR ASN.1 WITHOUT MODIFYING THE BASIC COMPILER AND CODE GENERATOR

Inventor #1: Mark Clark
Address: 2986 Little Rock Drive, San Jose, California 95133
Citizenship: USA

Inventor #2: Mark Skrzynski
Address: 1800 42nd Avenue, Capitola, California 95010
Citizenship: USA

Inventor #3: Florian Trinkwalder
Address: 530 Mansion Court #301, Santa Clara, CA 95054
Citizenship: Germany

METHOD FOR ADDING EXTENSIONS TO THE GRAMMER FOR ASN.1 WITHOUT MODIFYING THE BASIC COMPILER AND CODE GENERATOR

RESERVATION OF COPYRIGHT

BACKGROUND OF THE INVENTION

FIELD OF THE INVENTION

The present invention relates to data coding and, in particular, to an improved method for compiling grammar extensions to data codes.

DESCRIPTION OF THE RELATED ART

The Abstract Syntax Notation One (ASN.1) is a notation for specifying data structures at a relatively high level of abstraction. In particular, ASN.1 defines a data structure protocol for describing messages to be exchanged between distributed computer systems. ASN.1 is used to generate programming language code that forms the core of a wide variety of messaging systems applications. For example, H.323 based systems use ASN.1 coded messages for signaling between H.323 endpoints, servers, gateways, and gatekeepers.

The ASN.1 language has evolved over time to include the ITU-T standards X.680 (Specification of Basic Notation), X.681 (Information Object Specification), X.682 (Constraint Specification), and X.683 (Parameterization of ASN.1 Specifications), all of which are hereby incorporated by reference in their entireties as if fully set forth herein.

The basic ASN.1 compiler supports only X.680 grammar constructs. Typically, each extension has been supported through the development of an entirely new compiler. However, writing a compiler is relatively expensive and

time consuming. Moreover, in certain instances, it is not possible because the code generated is proprietary. When new extensions of the grammar are defined, but the available compiler does not support them, the user cannot take advantage of the new features.

Typically, this has required the user to purchase a new compiler which, again, is relatively expensive. Alternatively, the user can manually convert the new grammar constructs into those that are supported by the old compiler.

SUMMARY OF THE INVENTION

These and other problems in the prior art are overcome in large part by a system and method according to the present invention. A computer system is provided including a software program that reads source files that contain extended grammar constructs of ASN.1 and generates a new file that is a translation of the input file without the extended constructs. The new file contains only the basic ASN.1 constructs. The new file may then be compiled using a standard ASN.1 compiler.

A computer system according to an implementation of the invention implements a translator and a compiler for compiling a machine readable language. When a source file is provided, the system performs a line-by-line compare to determine if the file contains extended grammar constructs of the language. If not, then the source file is compiled. If so, however, then the source file is input to the translator to translate into basic grammar constructs.

BRIEF DESCRIPTION OF THE DRAWINGS

A better understanding of the invention is obtained when the following detailed description is considered in conjunction with the following drawings in which:

FIG. 1 is a diagram illustrating a system according to an implementation of the invention;

FIG. 2 is a diagram schematically illustrating operation of an embodiment of the invention;

FIG. 3 is a diagram illustrating a translator according to an embodiment

of the invention; and

FIG. 4 is a flowchart illustrating operation of an embodiment of the invention.

DETAILED DESCRIPTION OF THE INVENTION

FIGS. 1- 4 illustrate an improved system and method for compiling ASN.1 grammar constructs. A computer system is provided including a software program that reads source files that contain extended grammar constructs of ASN.1 and generates a new file that is a translation of the input file without the extended constructs. The new file contains only the basic ASN.1 constructs. The new file is then compiled using a standard ASN.1 compiler.

An exemplary processing system according to an implementation of the invention is shown in FIG. 1. As will be described in greater detail below, the system 102 may be embodied as an H.323 terminal or a general purpose computer system. In particular, the system may be embodied as a personal computer, such as an x86 compatible personal computer or an Apple Macintosh G4. The computer 102 includes a processor 11 adapted to implement computer code according to the present invention. As shown in the figure, a compiler 106 and a translator or precompiler 102 are resident.

Operation of an embodiment of the present invention is illustrated schematically with reference to FIG. 2. A source file 100 includes extended grammar constructs of ASN.1. Thus, the extended source file may include grammar constructs from X.68x (e.g., X.680, X.681, etc.). The extended source file 100 is input to the translator unit 102. The translator unit 102 may include one or more databases that store conversion tables of the extended and non-extended ASN.1 grammar constructs. The translator unit 102 outputs a basic source file 104. The basic source file 104 includes only grammar constructs supported by the basic compiler, such as only X.680 constructs. The basic source file 104 is input to the compiler unit 106. The compiler unit 106 may be any standard compiler that supports only the basic constructs. However, since the source file 100 has been translated into the source file 104, the compiler 106 is able to produce a compiled executable file

108.

It is noted that the precompiler or translator 102 may be implemented having differing or even multiple configurable levels of precompilation. For example, the compiler 106 may support X.680 and X.681 constructs; in that case, the precompiler 102 need only implement X.682 and higher precompilation.

An exemplary translator or precompiler 102 is illustrated with reference to FIG. 3. As shown, the translator 102 includes a plurality of lookup tables 404a-404n. The lookup tables 404a-404n include entries for particular grammar extensions 406a-406n, and their corresponding basic grammar constructs 408a-408n. As will be explained in greater detail below, the translator 102 accesses the appropriate lookup table for the basic grammar constructs once the particular extension is identified.

In particular, FIG. 4 illustrates a flowchart according to a specific embodiment of the invention. In a step 502, the system reads a source file 100. In a step 504, the system determines whether the file contains the basic grammar or whether it also contains extended grammar constructs. For example, the system may access one or more of the lookup tables 404a-404n and make a line-by-line comparison of the source file to make this determination. If the file contains only basic grammar constructs, then in a step 506, the file 100 is compiled using the compiler 106. Otherwise, in a step 508, the system accesses the lookup tables in the translator 102 and performs the appropriate translation of the grammar into the basic grammar constructs. In a step 510, the system generates the new basic source file, and may save the file to disk in a step 512. Finally, in step 506, the new source file is compiled using the compiler 106.

The following examples illustrate files that are translated from extended grammar constructs to the basic grammar constructs only:

EXAMPLE 1:

With Extensions

```
BEGIN

 SIGNED { ToBeSigned } ::= SEQUENCE {
   toBeSigned      ToBeSigned,
   algorithmOID  OBJECT IDENTIFIER,
   signature       BIT STRING
 }

 H235CertificateSignature      ::=SEQUENCE
 {
      argument    Argument,
      signature   SIGNED { EncodedReturnSig },
      ...
 }

 Argument ::= INTEGER
 EncodedReturnSig ::= NULL

       END  -- of Test-ASN
```

Without Extensions

```
Test-ASN

 DEFINITIONS AUTOMATIC TAGS ::=

 BEGIN

 H235CertificateSignature      ::=SEQUENCE
 {
      argument    Argument,
      signature   SEQUENCE {
                            toBeSigned     EncodedReturnSig,
                            algorithmOID OBJECT IDENTIFIER,
                            signature      BIT STRING
                  },
      ...
 }

 Argument ::= INTEGER
 EncodedReturnSig ::= NULL

END  -- of Test-ASN
```

EXAMPLE 2

With extensions

```
Test-ASN

 DEFINITIONS AUTOMATIC TAGS ::=

 BEGIN

    TESTOPERATION ::= CLASS
```

```
        {
              &arguments  Arguments OPTIONAL
        }
        WITH SYNTAX
        {
              [ARGUMENTS                &arguments]
        }


        Arguments ::= CHOICE
        {
              argument1    INTEGER,
              argument2    INTEGER,
              argument3    INTEGER,
              argument4    INTEGER,
              argument5    INTEGER,
              argument6    INTEGER,
        }


        myTestOperation TESTOPERATION ::=
        {
              ARGUMENTS
        }

    TestOperationSet TESTOPERATION ::= {myTestOperation}


    myTest ::= SEQUENCE
    {
        arguments TESTOPERATION.&arguments({TestOperationSet})
    }

 END  -- of Test-ASN
```

Without extensions

```
Test-ASN
 DEFINITIONS AUTOMATIC TAGS ::=

 BEGIN

      myTest ::= SEQUENCE
    {
    arguments Arguments
    }

      Arguments ::= CHOICE
      {
            argument1    INTEGER,
            argument2    INTEGER,
            argument3    INTEGER,
            argument4    INTEGER,
            argument5    INTEGER,
            argument6    INTEGER,
      }

 END  -- of Test-ASN
```

The invention described in the above detailed description is not intended to be limited to the specific form set forth herein, but is intended to cover such alternatives, modifications and equivalents as can reasonably be included within the spirit and scope of the appended claims.